**递归与回溯**

**一、递归简介**

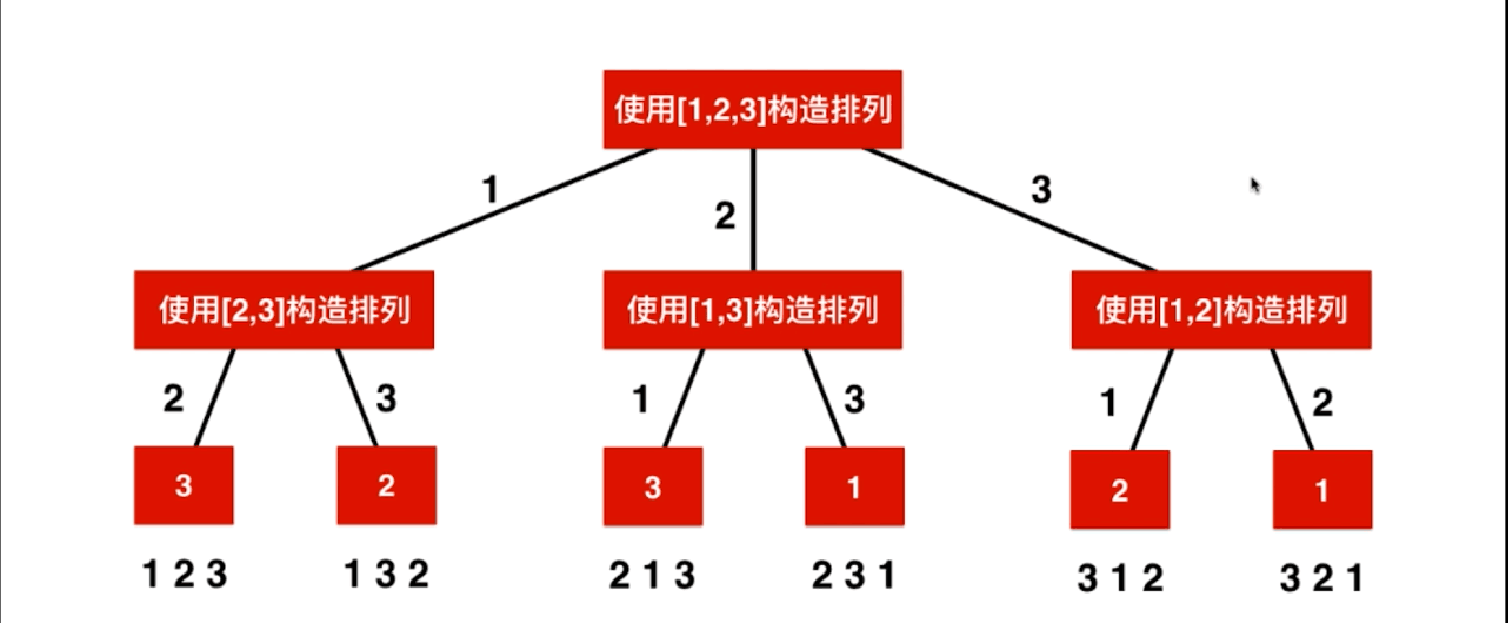
1. 定义：自己调用自己 ==> 头接屁股一直转，实现是用栈（先进后出）
   1. Void printA(){

printA();

}

注：这样会造成死循环，必须要有递归停止的条件

1. 我们先来实现一个小的递归
   1. 实现阶乘？？ 5！
      1. 5! = 5x4x3x2x1
      2. 分解：
         1. 第一步：5x4
         2. 第二步：4x3
         3. 第三步：3x2
         4. 第四步：2x1
         5. 返回即我们递归退出的条件，就是n==1退出
   2. 代码：
      1. long fac(int n){
      2. if(n==1)
      3. return 1;
      4. return fac(n-1) \* n;
      5. }
   3. 分步进行解释
      1. 先放进栈中
      2. 第一：n=5进入函数，判断n=1 != 1 所以执行fac(n-1)\*5
      3. 第二：在n=4进入函数，n-1 != 1 所以执行fac(n-1)\*4
      4. ……………………
      5. 最后：当n=2时，n-1 == 1我们返回1回去
      6. 出栈
      7. 第一：n=2时， n-1 == 1， 所以返回 1 x 2 即 fac(2) = 2
      8. 第二：n=3时， n-1 == 2， 所以返回 fac(2) x 3 即 2x3 = fac(3) = 6
      9. ………………
      10. 最后：n == 5， n-1 == 4， 所以返回 fac(4) x 5 即 自己算
2. 总结
   1. 优点：代码简洁
   2. 缺点：不好理解、耗费资源、运行时间长
3. **递归与回溯**
4. 含义，我们提到递归一般都会提到回溯，其实说来就是暴力解法，挨个进行尝试
5. 我们用一个小实例来领会一下（回溯）[1, 2, 3]来进行全排列（leetcode 46题）
   * 1. [1, 2, 3], [1, 3, 2], [2, 1, 3], [2, 3, 1],[3, 2, 1], [3, 1, 2]
     2. 实现步骤



* + 1. 利用树形结构：
       1. 第一步：加入我们选 1
       2. 第二步：我们从剩下的 2，3 中任选取 2
       3. 第三步：第二步选取 2 ，所以剩下 3 一个数结束本次
       4. 第四步：我们第二步选取的是2， 但是我们还有一个选择3，所以我们需要回溯到第二部再继续递归
       5. 以此类推，直到所有的结束
    2. 伪代码：Perm(nums[0, n-1]) = {这个数} + Perm(nums[0, n-1] - 这个数)
    3. 但是相对于我们求5！不同，我们的元素再[1, 2, 3]使用了，但是在[1, 3, 2]还是可以用的，所以我们要记得释放掉

1. 代码的实现（代码用C++实现）自行体会一下吧

class Solution {

private:

vector< vector<int> > res; //用来存放结果

vector<bool> used; //判断当前数是否被访问过了

//nums 是我们要处理的数组， index 表示我们要处理的是第几个数， p用来存放一组结果

void perm(const vector<int>& nums, int index, vector<int>& p){

//如果我们处理的已经是最后一个就可以放进数组里面了

if(index == nums.size()){

res.push\_back(p);

}

for(int i = 0; i < nums.size(); i++){

// 如果该数字没被用过则推入p数组中去

if(!used[i]){

p.push\_back(nums[i]);

used[i] = true;

perm(nums, index+1, p);

//记得回溯回去哦

p.pop\_back();

used[i] = false;

}

}

}

public:

vector<vector<int>> permute(vector<int>& nums) {

res.clear();

if(nums.size() == 0){

return res;

}

used = vector<bool>(nums.size(), false);

vector<int> p;

//如果我们传进去的数组是空，则返回空

perm(nums, 0, p);

return res;

}

};